AF

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE
BEFORE THE BOARD OF PATENT APPEALS AND INTERFERENCES

APPLICANT(s):	Toth et al.		
SERIAL NO.:	10/019,330	ART UNIT:	2143
FILING DATE:	03/07/2002	EXAMINER:	Avellino, Joseph E.
TITLE:	SERVER TERMINAL(S) SESSION MANAGEMENT INVOLVING ASSIGNING GROUPS OF SESSIONS TO THREADS		
ATTORNEY DOCKET NO.:	442-010757-US (PAR)		

Board of Patent Appeals and Interferences
United States Patent and Trademark Office
P.O. Box 1450
Alexandria, VA 22313-1450

APPELLANTS' BRIEF
(37 C.F.R. § 41.37)

This is an appeal from the final rejection of the claims in the above-identified application. A Notice of Appeal was mailed on December 13, 2005. The fees required under 37 C.F.R. § 41.20 are being submitted herewith.

I. REAL PARTY IN INTEREST

The real party in interest in this Appeal is the Assignee, Nokia Corporation, Espoo, Finland.

II. RELATED APPEALS AND INTERFERENCES

There are no related appeals or interferences regarding this application.

III. STATUS OF CLAIMS

Claims 1-22 are pending in the application.

Claims 1-22 have been finally rejected.

The claims on appeal are 1-22.

IV. STATUS OF AMENDMENTS

No amendments to the claims were made subsequent to the final rejection.

V. SUMMARY OF CLAIMED SUBJECT MATTER

Claim 1 recites a method of managing a plurality of sessions (66) the sessions being between a plurality of terminals (2) and

a server (20) having a plurality of threads (74) (Figures 2 and 4; page 12, lines 14-21). The method comprising grouping the sessions into a plurality of groups (72) (page 12, line 22 - page 13, line 9) and assigning a thread (74) to each group (72) of sessions so that the assigned thread (74) only handles the events of that group of sessions (page 12, lines 26-27).

Claim 18 recites a server (20) for managing a plurality of sessions with a plurality of terminal (2) (Figure 2; page 12, lines 14-21). The server (20) comprising a plurality of threads (74), grouping means to group the sessions into a plurality of groups and assigning means to assign a thread to each group of sessions so that the assigned thread (74) only handles the events of that group (72) of sessions (page 12, line 22 - page 13, line 9).

Claim 21 recites a communications system comprising a server (20) and a plurality of terminals (2) the server (20) and the terminals (2) conducting a plurality of sessions (66) the server comprising a plurality of threads (74) (Figures 1, 2 and 4; page 10, lines 21-22; page 12, lines 14-21), grouping means to group the sessions into a plurality of groups and assigning means to assign at least one thread to each group of sessions so that the assigned thread (74) only handles the events of that group (72) of sessions (page 12, line 22 - page 13, line 9).

Claim 22 recites a computer program product for managing a plurality of sessions (66) the sessions being between a plurality of terminals (2) and a server (20) having a plurality of threads (74) (Figures 2 and 4; page 12, lines 14-21). The computer program product comprising computer readable program means for grouping the sessions (66) into a plurality of groups

(72) and computer readable program means for assigning a thread to each group (72) of sessions so that the assigned thread (74) only handles the events of that group (72) of sessions(page 12, line 22 - page 13, line 9).

VI. GROUNDS OF REJECTION TO BE REVIEWED ON APPEAL

The issues presented for review are whether:

A. claims 1-4, 9-11, 15, 18, 21 and 22 are patentable under 35 U.S.C. 102(e) over Bayeh et al., U.S. Patent No. 6,098,093 ("Bayeh"); and

B. claims 5-8, 12-14 and 17 are patentable under 35 U.S.C. 103(a) over Bayeh.

VII. ARGUMENT

A. 35 U.S.C. 102(e)

1. Claim 1

Claim 1 is patentable under 35 U.S.C. 102(e) over Bayeh. Claim 1 of the present application recites a server having a plurality of threads, grouping the sessions into a plurality of groups and assigning a thread to each group of sessions so that the assigned thread only handles the events of that group of sessions. Bayeh does not disclose or suggest these features.

Bayeh discloses a clustered server environment (Col. 8, L. 42-45; Fig. 3). A web server (60) may be connected to any number of other Web servers (62, 64). Clustering multiple servers in

this way provides for increased capacity with which HTTP requests at a Web site can be processed. (Col. 8, L. 45-49).

Session services support for the clustered server environment is provided in Bayeh by providing session services management features in a plug-in servlet engine, which is executable code that extends the functionality of the Web server. One of the servlet engines will be installed on each Web server that will participate in the session management solution of Bayeh. (Col. 8, L. 59-66). One of the servlets will be configured to function as a session server and the others will be configured to function as session clients (Col. 8, L. 66 - Col. 9, L. 2).

In Bayeh there is no control mechanism for a thread to handle the requests of a particular session. Instead, any server can handle the requests of a particular session and a mechanism is present to provide the session object to that server. There is no **assignment of a thread to a session** because sessions are not tied to any particular server. Furthermore, because there is no assignment of a thread to a session, there cannot be assignment of a thread to a **group of sessions**. Therefore, claim 1 cannot be anticipated by Bayeh.

The concepts underlying the claimed features of Appellant's invention are quite different from Bayeh. Bayeh relates to a technique, system, and computer program for maintaining session-related state information in a scalable, clustered network environment (Col. 1, L. 8 to 10). Bayeh is concerned with providing the necessary session information (state information) to enable session requests to be handled. Sessions are identified by session IDs and state information, describing the

state of a session, is stored on the server using a session object. This object is created when a new client session begins, and is kept for the duration of the session. The object stores information about the transactions occurring between the client and the server (Col. 4, L. 5 to 9) and is shared among servlets (Col. 4, L. 23). In terms of organizing sessions, Bayeh is explicit in stating that [w]hen session services are provided at these clustered servers, the group of sessions can logically be thought of as a *"pool, or cluster, of sessions. Since the Session Tracking facility in the Java Web Server Toolkit is only valid within the scope of a single Web server, the pool of sessions cannot be properly maintained among a group of clustered Web servers such as this."* (Col. 4, L. 59-65).

For example, if a client request is received at one server, and that server maintains information about the on-going session, there is no way for this version of the session information to be made available to a different server in the cluster if the next request from this client goes to a different server. (Col. 4, L. 59 to Col. 5, L. 3).

Thus, Bayeh teaches that sessions are identified by session IDs, session state information is maintained as session objects, the objects are shared among servlets which handle session requests, and that in a clustered server environment, one server might handle a request for a particular session and then another server might handle a subsequent request for that session. In other words, the particular session is not associated with a particular server because different servers might handle the requests for that session.

Bayeh is concerned with the discovery and retrieval of the appropriate session object in handling a request. Because Bayeh is a clustered server environment, when a request arrives which is part of a session, since the environment is not arranged to route the request to a particular server to deal with the request, the server receiving the request needs to be provided with the session object which is relevant to that session. Therefore, "[t]*he servlet engine configured as a session server further comprises a subprocess for maintaining a plurality of session objects; a subprocess for locating one of the session objects in response to a request from one of the servlets or one of the session clients; and a subprocess for returning the located session object in response to the request.*" (Col. 5, L. 65 - Col. 6, L. 4). This is not the same as in Appellant's invention, where the sessions are grouped into a plurality of groups and a "thread" is assigned to each "group" of sessions, and the **assigned thread only handles the events of the group of sessions**.

Bayeh also discloses, a load-balancing host (59) functions as a type of front-end processor to these servers, receiving client requests (100, 101, 102) and then routing those requests (shown here as 110, 111, 112) to a server selected according to policies implemented in the load-balancing host software. Note that the requests (110, 111, 112) are shown being sent to specific Web servers: this is merely an example of a possible outcome of the load balancing process. Load-balancing techniques are known in the art, and do not form part of the present invention. (Col. 8, L. 49-58.) Load balancing happens on an arbitrary basis as far as sessions are concerned which of course is not significant because the whole point of Bayeh is

that irrespective of which server receives a request, the session object can be made available.

The nature of the servlets, and the fact that they are not assigned to any particular session is underlined when one considers subsequent parts of Bayeh:

> *If a servlet is written not to use session-related information, the servlet will perform its specific processing, eventually finishing and returning its results to the server through the response object with which it was invoked. While the servlet may have been invoked with a valid session ID present in the request object, it will not have made use of that session. (Col. 10, L. 54-60.)*

and,

> *If session services are required for this servlet's application, then the session services features of the present invention are used. According to the present invention, a servlet in this scenario will include code to get the session object for this session, and update that object to reflect session information related to the servlet's processing. When the servlet processing is finished, the session object is returned to the session pool, where it can be accessed for subsequent transactions with this servlet or a different servlet in the clustered environment. In this way, the state of the session can be communicated among the clustered servers and their servlets. (Col. 10, L. 64 - Col. 11, L. 8.)*

The features of Appellant's invention as recited in claim 1 are different from Bayeh, because in Appellant's invention the

sessions are between a plurality of terminals and a server. The server has a plurality of threads. The sessions are grouped into a plurality of groups and a "thread" is assigned to each "group" of sessions. Each assigned thread only handles the events of the group of sessions. This is not disclosed or suggested by Bayeh. Thus, claim 1 cannot be anticipated by Bayeh.

Further, the Examiner's argument that the "servlets" of Bayeh are the same as the "threads" claimed by Appellant is incorrect. "Servlets" and "threads" are not the same. A servlet is defined as an "applet that runs on a server." (Newton's Telecom Dictionary, 18th Edition by Harry Newton (2002), page 662; Attached hereto as Exhibit A). The term usually refers to a Java applet that runs within a Web server environment. An "applet" is a mini-program "that can be downloaded quickly and used by any computer equipped with a Java-capable browser." (Newton's Telecom Dictionary, 18th Edition by Harry Newton (2002) at page 57; Attached hereto as Exhibit B). A "thread" on the other hand is defined as "a sequence of computing instructions that makes up a process or program." (Newton's Telecom Dictionary, 18th Edition by Harry Newton (2002) at page 747, definition 1; Attached hereto as Exhibit C). This difference between a servlet and a thread is supported by Bayeh itself, for example Bayeh described the "plug-in servlet engine" as "executable code that extends the functionality of the Web server" (Col. 8, L. 63-64). Also, at column 12, lines 46-48 Bayeh describes the use of threads in a servlet. Servlets are able to handle concurrent requests, which is done by using multiple threads. A servlet is a set of machine instructions

and individual ones of the instructions can be executed by different threads.

A "thread", is essentially a placeholder information associated with a single use of a program that can handle multiple concurrent users. From the program's point of view, a thread is the resource needed to serve one individual user or a particular service request. If multiple users are using the program or concurrent requests from other programs occur, a thread is created and maintained for each of them. A program can be single-threaded or multi-threaded. A single-threaded application program insists that only a single instruction can be executed at a given time. All the instructions must be executed in an exact sequence, from the beginning to end. For example, a single-threaded Internet experience might involve your accessing a Web-based sever that would accept your request to establish a session, and would accept and serve your request for information. During this period of time, a tightly choreographed series of steps would take place in exact sequence, and no requests from other clients would be accepted until your request was satisfied. In other words, a single-threaded program must follow a single line of logic in a very rigid manner.

A multi-threaded process has multiple threads, each executing independently and each perhaps executing on separate processors within one or multiple computers. A multi-threaded program has multiple points of execution (one per thread) and, therefore can perform multiple tasks associated with multiple processes and multiple programs supporting multiple users at any given time. As each task associated with each task associated with each

process supporting each program and each user is completed, the thread for that task is resumed at the same point it had been interrupted, much as though it had been book-marked. As multiple instruction sets can be executed concurrently, the throughput and speed of running the program is much improved. In other words, a multi-threaded program, if running on a computer with multiple processors, will run much faster than a single-threaded program running on a single processor machine.

Thus, a "servlet" and "thread" are not the same and there is no teaching in Bayeh that servlets are interchangeable with threads. In fact, in column 12 lines 46-51, Bayeh discusses them as separate entities, "servlet threads" and the "servlet process". Therefore, claim 1 is patentable over Bayeh.

2. Claim 4

Claim 4 is dependent on claim 1 and is allowable at least for the reasons noted before with respect to claim 1. Further, claim 4 recites one group is provided for each thread so that there are equal numbers of groups and threads. Bayeh does not disclose providing one group for each thread so there are an equal number of groups and threads.

Bayeh discloses providing session services management features in a plug-in servlet engine, which is executable code that extends the functionality of the Web server. One of these servlet engines will be installed on each Web server that will participate in this session-management solution (Col. 8, L. 59-66).

The Examiner equates that "servlet engines" are the same as "threads" and the each Web server of Bayeh is a group in

arriving at his conclusion there are an equal numbers of groups and threads in Bayeh. However, the "servlet engines" and "threads" are not the same. A servlet is defined as an "applet that runs on a server." (Newton's Telecom Dictionary, 18th Edition by Harry Newton (2002), page 662; Attached hereto as Exhibit A). The term usually refers to a Java applet that runs within a Web server environment. An "applet" is a mini-program "that can be downloaded quickly and used by any computer equipped with a Java-capable browser." (Newton's Telecom Dictionary, 18th Edition by Harry Newton (2002) at page 57; Attached hereto as Exhibit B). A "thread" on the other hand is defined as "a sequence of computing instructions that makes up a process or program." (Newton's Telecom Dictionary, 18th Edition by Harry Newton (2002) at page 747, definition 1; Attached hereto as Exhibit C). This difference between a servlet and a thread is supported by Bayeh itself, for example Bayeh described the "plug-in servlet engine" as "executable code that extends the functionality of the Web server" (Col. 8, L. 63-64). Also, at column 12, lines 46-48 Bayeh describes the use of threads in a servlet. Servlets are able to handle concurrent requests which is done by using multiple threads. A servlet is a set of machine instructions and individual ones of the instructions can be executed by different threads. Thus, because threads are not the same as servlets there cannot be an equal number of threads and groups as recited in claim 4. In fact Bayeh discloses that if a "session is not valid, then step (430) generates events to notify the waiting threads" (Col. 122, L. 67 - Col. 13, L. 1) further evidencing that there is not one group provided for each thread. Therefore, claim 4 is patentable over Bayeh.

3. Claim 10

Claim 10 is dependent on claim 1 and is allowable at least for the reasons noted before with respect to claim 1. Further, claim 10 recites the sessions are grouped by a thread referred to as an acceptor thread. Bayeh does not disclose or suggest this feature.

The Examiner argues that the load balancing process at column 8, lines 42-58 of Bayeh disclose the grouping of the session as claimed in claim 10. This passage of Bayeh discloses a load-balancing host (59) functions as a type of front-end processor to these servers, receiving client requests (100, 101, 102) and then routing those requests (shown here as 110, 111, 112) to a server selected according to policies implemented in the load-balancing host software. Note that the requests (110, 111, 112) are shown being sent to specific Web servers: this is merely an example of a possible outcome of the load balancing process. Load-balancing techniques are known in the art, and do not form part of the present invention. (Col. 8, L. 49-58.)

Load balancing happens on an arbitrary basis as far as sessions are concerned which of course is not significant because the whole point of Bayeh is that irrespective of which server receives a request, the session object can be made available. The Load balancing software of Bayeh does not group the sessions as claimed by Appellant. Load balancing software is merely for distributing processing and communications activity evenly across a computer network so that no single device is overwhelmed. If one server starts to get swamped, requests are forwarded to another server with more capacity. Thus, load balancing of Bayeh is an overflow tool that passes requests from

one server to another server when the requests are to numerous for handling by a single server. There is no grouping involved with load balancing. Thus, claim 10 cannot be anticipated by Bayeh.

4. Claim 18

Claim 18 is patentable under 35 U.S.C. 102(e) over Bayeh. Claim 18 of the present application recites the server comprising a plurality of threads, grouping means to group the sessions into a plurality of groups and assigning means to assign a thread to each group of sessions so that the assigned thread only handles the events of that group of sessions. Bayeh does not disclose or suggest these features.

In Bayeh there is no control mechanism for a thread to handle the requests of a particular session. Instead, any server can handle the requests of a particular session and a mechanism is present to provide the session object to that server. There is no **assignment of a thread to a session** because sessions are not tied to any particular server. Furthermore, because there is no assignment of a thread to a session, there cannot be assignment of a thread to a **group of sessions**. Therefore, claim 18 cannot be anticipated by Bayeh.

The concepts underlying the claimed features of Appellant's invention are quite different from Bayeh. Bayeh relates to a technique, system, and computer program for maintaining session-related state information in a scalable, clustered network environment (Col. 1, L. 8 to 10). Bayeh is concerned with providing the necessary session information (state information) to enable session requests to be handled. Sessions are

identified by session IDs and state information, describing the state of a session, is stored on the server using a session object. This object is created when a new client session begins, and is kept for the duration of the session. The object stores information about the transactions occurring between the client and the server (Col. 4, L. 5 to 9) and is shared among servlets (Col. 4, L. 23). In terms of organizing sessions, Bayeh is explicit in stating that [w]hen session services are provided at these clustered servers, the group of sessions can logically be thought of as a *"pool, or cluster, of sessions. Since the Session Tracking facility in the Java Web Server Toolkit is only valid within the scope of a single Web server, the pool of sessions cannot be properly maintained among a group of clustered Web servers such as this."* (Col. 4, L. 59-65).

For example, if a client request is received at one server, and that server maintains information about the on-going session, there is no way for this version of the session information to be made available to a different server in the cluster if the next request from this client goes to a different server. (Col. 4, L. 59 to Col. 5, L. 3).

Thus, Bayeh teaches that sessions are identified by session IDs, session state information is maintained as session objects, the objects are shared among servlets which handle session requests, and that in a clustered server environment, one server might handle a request for a particular session and then another server might handle a subsequent request for that session. In other words, the particular session is not associated with a particular server because different servers might handle the requests for that session.

Bayeh is concerned with the discovery and retrieval of the appropriate session object in handling a request. Because Bayeh is a clustered server environment, when a request arrives which is part of a session, since the environment is not arranged to route the request to a particular server to deal with the request, the server receiving the request needs to be provided with the session object which is relevant to that session. Therefore, "[t]*he servlet engine configured as a session server further comprises a subprocess for maintaining a plurality of session objects; a subprocess for locating one of the session objects in response to a request from one of the servlets or one of the session clients; and a subprocess for returning the located session object in response to the request.*" (Col. 5, L. 65 - Col. 6, L. 4). This is not the same as in Appellant's invention, where the sessions are grouped into a plurality of groups and a "thread" is assigned to each "group" of sessions, and the **assigned thread only handles the events of the group of sessions**.

Bayeh also discloses, a load-balancing host (59) functions as a type of front-end processor to these servers, receiving client requests (100, 101, 102) and then routing those requests (shown here as 110, 111, 112) to a server selected according to policies implemented in the load-balancing host software. Note that the requests (110, 111, 112) are shown being sent to specific Web servers: this is merely an example of a possible outcome of the load balancing process. Load-balancing techniques are known in the art, and do not form part of the present invention. (Col. 8, L. 49-58.) Load balancing happens on an arbitrary basis as far as sessions are concerned which of course is not significant because the whole point of Bayeh is

that irrespective of which server receives a request, the session object can be made available.

The nature of the servlets, and the fact that they are not assigned to any particular session is underlined when one considers subsequent parts of Bayeh:

> *If a servlet is written not to use session-related information, the servlet will perform its specific processing, eventually finishing and returning its results to the server through the response object with which it was invoked. While the servlet may have been invoked with a valid session ID present in the request object, it will not have made use of that session. (Col. 10, L. 54-60.)*

and

> *If session services are required for this servlet's application, then the session services features of the present invention are used. According to the present invention, a servlet in this scenario will include code to get the session object for this session, and update that object to reflect session information related to the servlet's processing. When the servlet processing is finished, the session object is returned to the session pool, where it can be accessed for subsequent transactions with this servlet or a different servlet in the clustered environment. In this way, the state of the session can be communicated among the clustered servers and their servlets. (Col. 10, L. 64 - Col. 11, L. 8.)*

The features of Appellant's invention as recited in claim 18 are different from Bayeh, because in Appellant's invention the

sessions are between a plurality of terminals and a server. The server has a plurality of threads. The sessions are grouped into a plurality of groups and a "thread" is assigned to each "group" of sessions. Each assigned thread only handles the events of the group of sessions. This is not disclosed or suggested by Bayeh. Thus, claim 18 cannot be anticipated by Bayeh.

Further, the Examiner's argument that the "servlets" of Bayeh are the same as the "threads" claimed by Appellant in incorrect. "Servlets" and "threads" are not the same. A servlet is defined as an "applet that runs on a server." (Newton's Telecom Dictionary, 18th Edition by Harry Newton (2002), page 662; Attached hereto as Exhibit A). The term usually refers to a Java applet that runs within a Web server environment. An "applet" is a mini-program "that can be downloaded quickly and used by any computer equipped with a Java-capable browser." (Newton's Telecom Dictionary, 18th Edition by Harry Newton (2002) at page 57; Attached hereto as Exhibit B). A "thread" on the other hand is defined as "a sequence of computing instructions that makes up a process or program." (Newton's Telecom Dictionary, 18th Edition by Harry Newton (2002) at page 747, definition 1; Attached hereto as Exhibit C). This difference between a servlet and a thread is supported by Bayeh itself, for example Bayeh described the "plug-in servlet engine" as "executable code that extends the functionality of the Web server" (Col. 8, L. 63-64). Also, at column 12, lines 46-48 Bayeh describes the use of threads in a servlet. Servlets are able to handle concurrent requests which is done by using multiple threads. A servlet is a set of machine instructions

and individual ones of the instructions can be executed by different threads.

A "thread", is essentially a placeholder information associated with a single use of a program that can handle multiple concurrent users. From the program's point of view, a thread is the resource needed to serve one individual user or a particular service request. If multiple users are using the program or concurrent requests from other programs occur, a thread is created and maintained for each of them. A program can be single-threaded or multi-threaded. A single-threaded application program insists that only a single instruction can be executed at a given time. All the instructions must be executed in an exact sequence, from the beginning to end. For example, a single-threaded Internet experience might involve your accessing a Web-based sever that would accept your request to establish a session, and would accept and serve your request for information. During this period of time, a tightly choreographed series of steps would take place in exact sequence, and no requests from other clients would be accepted until your request was satisfied. In other words, a single-threaded program must follow a single line of logic in a very rigid manner.

A multi-threaded process has multiple threads, each executing independently and each perhaps executing on separate processors within one or multiple computers. A multi-threaded program has multiple points of execution (one per thread) and, therefore can perform multiple tasks associated with multiple processes and multiple programs supporting multiple users at any given time. As each task associated with each task associated with each

process supporting each program and each user is completed, the thread for that task is resumed at the same point it had been interrupted, much as though it had been book-marked. As multiple instruction sets can be executed concurrently, the throughput and speed of running the program is much improved. In other words, a multi-threaded program, if running on a computer with multiple processors, will run much faster than a single-threaded program running on a single processor machine.

Thus, a "servlet" and "thread" are not the same and there is no teaching in Bayeh that servlets are interchangeable with threads. In fact, in column 12 lines 46-51, Bayeh discusses them as separate entities, "servlet threads" and the "servlet process". Therefore claim 18 is patentable over Bayeh.

5. Claim 21

Claim 21 is patentable under 35 U.S.C. 102(e) over Bayeh. Claim 21 of the present application recites a server and a plurality of terminals the server and the terminals conducting a plurality of sessions the server comprising a plurality of threads, grouping means to group the sessions into a plurality of groups and assigning means to assign at least one thread to each group of sessions so that the assigned thread only handles the events of that group of sessions. Bayeh does not disclose or suggest the server having a plurality of threads, grouping the sessions into a plurality of groups and assigning at least one thread to each group of sessions so that the assigned thread only handles the events of that group of sessions.

As discussed above, in Bayeh there is no control mechanism for a thread to handle the requests of a particular session. Instead,

any server can handle the requests of a particular session and a mechanism is present to provide the session object to that server. There is no **assignment of a thread to a session** because sessions are not tied to any particular server. Furthermore, because there is no assignment of a thread to a session, there cannot be assignment of a thread to a **group of sessions**. Therefore, claim 21 cannot be anticipated by Bayeh.

The concepts underlying the claimed features of Appellant's invention are quite different from Bayeh. Bayeh relates to a technique, system, and computer program for maintaining session-related state information in a scalable, clustered network environment (Col. 1, L. 8 to 10). Bayeh is concerned with providing the necessary session information (state information) to enable session requests to be handled. Sessions are identified by session IDs and state information, describing the state of a session, is stored on the server using a session object. This object is created when a new client session begins, and is kept for the duration of the session. The object stores information about the transactions occurring between the client and the server (Col. 4, L. 5 to 9) and is shared among servlets (Col. 4, L. 23). In terms of organizing sessions, Bayeh is explicit in stating that [w]hen session services are provided at these clustered servers, the group of sessions can logically be thought of as a *"pool, or cluster, of sessions. Since the Session Tracking facility in the Java Web Server Toolkit is only valid within the scope of a single Web server, the pool of sessions cannot be properly maintained among a group of clustered Web servers such as this."* (Col. 4, L. 59-65).

For example, if a client request is received at one server, and that server maintains information about the on-going session, there is no way for this version of the session information to be made available to a different server in the cluster if the next request from this client goes to a different server. (Col. 4, L. 59 to Col. 5, L. 3).

Thus, Bayeh teaches that sessions are identified by session IDs, session state information is maintained as session objects, the objects are shared among servlets which handle session requests, and that in a clustered server environment, one server might handle a request for a particular session and then another server might handle a subsequent request for that session. In other words, the particular session is not associated with a particular server because different servers might handle the requests for that session.

Bayeh is concerned with the discovery and retrieval of the appropriate session object in handling a request. Because Bayeh is a clustered server environment, when a request arrives which is part of a session, since the environment is not arranged to route the request to a particular server to deal with the request, the server receiving the request needs to be provided with the session object which is relevant to that session. Therefore, "[t]*he servlet engine configured as a session server further comprises a subprocess for maintaining a plurality of session objects; a subprocess for locating one of the session objects in response to a request from one of the servlets or one of the session clients; and a subprocess for returning the located session object in response to the request.*" (Col. 5, L. 65 - Col. 6, L. 4). This is not the same as in Appellant's

invention, where the sessions are grouped into a plurality of groups and a "thread" is assigned to each "group" of sessions, and the **assigned thread only handles the events of the group of sessions**.

Bayeh also discloses, a load-balancing host (59) functions as a type of front-end processor to these servers, receiving client requests (100, 101, 102) and then routing those requests (shown here as 110, 111, 112) to a server selected according to policies implemented in the load-balancing host software. Note that the requests (110, 111, 112) are shown being sent to specific Web servers: this is merely an example of a possible outcome of the load balancing process. Load-balancing techniques are known in the art, and do not form part of the present invention. (Col. 8, L. 49-58.) Load balancing happens on an arbitrary basis as far as sessions are concerned which of course is not significant because the whole point of Bayeh is that irrespective of which server receives a request, the session object can be made available.

The nature of the servlets, and the fact that they are not assigned to any particular session is underlined when one considers subsequent parts of Bayeh:

> *If a servlet is written not to use session-related information, the servlet will perform its specific processing, eventually finishing and returning its results to the server through the response object with which it was invoked. While the servlet may have been invoked with a valid session ID present in the request object, it will not have made use of that session. (Col. 10, L. 54-60.)*

and

> *If session services are required for this servlet's application, then the session services features of the present invention are used. According to the present invention, a servlet in this scenario will include code to get the session object for this session, and update that object to reflect session information related to the servlet's processing. When the servlet processing is finished, the session object is returned to the session pool, where it can be accessed for subsequent transactions with this servlet or a different servlet in the clustered environment. In this way, the state of the session can be communicated among the clustered servers and their servlets. (Col. 10, L. 64 - Col. 11, L. 8.)*

The features of Appellant's invention as recited in claim 21 are different from Bayeh, because in Appellant's invention the sessions are between a plurality of terminals and a server. The server has a plurality of threads. The sessions are grouped into a plurality of groups and a "thread" is assigned to each "group" of sessions. Each assigned thread only handles the events of the group of sessions. This is not disclosed or suggested by Bayeh. Thus, claim 21 cannot be anticipated by Bayeh.

Further, the Examiner's argument that the "servlets" of Bayeh are the same as the "threads" claimed by Appellant in incorrect. "Servlets" and "threads" are not the same. A servlet is defined as an "applet that runs on a server." (Newton's Telecom Dictionary, 18th Edition by Harry Newton (2002), page 662; Attached hereto as Exhibit A). The term usually refers to a

Java applet that runs within a Web server environment. An "applet" is a mini-program "that can be downloaded quickly and used by any computer equipped with a Java-capable browser." (Newton's Telecom Dictionary, 18th Edition by Harry Newton (2002) at page 57; Attached hereto as Exhibit B). A "thread" on the other hand is defined as "a sequence of computing instructions that makes up a process or program." (Newton's Telecom Dictionary, 18th Edition by Harry Newton (2002) at page 747, definition 1; Attached hereto as Exhibit C). This difference between a servlet and a thread is supported by Bayeh itself, for example Bayeh described the "plug-in servlet engine" as "executable code that extends the functionality of the Web server" (Col. 8, L. 63-64). Also, at column 12, lines 46-48 Bayeh describes the use of threads in a servlet. Servlets are able to handle concurrent requests, which is done by using multiple threads. A servlet is a set of machine instructions and individual ones of the instructions can be executed by different threads.

A "thread", is essentially a placeholder information associated with a single use of a program that can handle multiple concurrent users. From the program's point of view, a thread is the resource needed to serve one individual user or a particular service request. If multiple users are using the program or concurrent requests from other programs occur, a thread is created and maintained for each of them. A program can be single-threaded or multi-threaded. A single-threaded application program insists that only a single instruction can be executed at a given time. All the instructions must be executed in an exact sequence, from the beginning to end. For example, a single-threaded Internet experience might involve your accessing

a Web-based sever that would accept your request to establish a session, and would accept and serve your request for information. During this period of time, a tightly choreographed series of steps would take place in exact sequence, and no requests from other clients would be accepted until your request was satisfied. In other words, a single-threaded program must follow a single line of logic in a very rigid manner.

A multi-threaded process has multiple threads, each executing independently and each perhaps executing on separate processors within one or multiple computers. A multi-threaded program has multiple points of execution (one per thread) and, therefore can perform multiple tasks associated with multiple processes and multiple programs supporting multiple users at any given time. As each task associated with each task associated with each process supporting each program and each user is completed, the thread for that task is resumed at the same point it had been interrupted, much as though it had been book-marked. As multiple instruction sets can be executed concurrently, the throughput and speed of running the program is much improved. In other words, a multi-threaded program, if running on a computer with multiple processors, will run much faster than a single-threaded program running on a single processor machine.

Thus, a "servlet" and "thread" are not the same and there is no teaching in Bayeh that servlets are interchangeable with threads. In fact, in column 12 lines 46-51, Bayeh discusses them as separate entities, "servlet threads" and the "servlet process". Therefore, claim 21 is patentable over Bayeh.

6. Claim 22

Claim 22 is patentable under 35 U.S.C. 102(e) over Bayeh. Claim 22 of the present application recites a server having a plurality of threads, computer readable program means for grouping the sessions into a plurality of groups and computer readable program means for assigning a thread to each group of sessions so that the assigned thread only handles the events of that group of sessions. Bayeh does not disclose or suggest the server having a plurality of threads, grouping the sessions into a plurality of groups and assigning a thread to each group of sessions so that the assigned thread only handles the events of that group of sessions.

Again, in Bayeh there is no control mechanism for a thread to handle the requests of a particular session. Instead, any server can handle the requests of a particular session and a mechanism is present to provide the session object to that server. There is no **assignment of a thread to a session** because sessions are not tied to any particular server. Furthermore, because there is no assignment of a thread to a session, there cannot be assignment of a thread to a **group of sessions**. Therefore, claim 22 cannot be anticipated by Bayeh.

The concepts underlying the claimed features of Appellant's invention are quite different from Bayeh. Bayeh relates to a technique, system, and computer program for maintaining session-related state information in a scalable, clustered network environment (Col. 1, L. 8 to 10). Bayeh is concerned with providing the necessary session information (state information) to enable session requests to be handled. Sessions are identified by session IDs and state information, describing the

state of a session, is stored on the server using a session object. This object is created when a new client session begins, and is kept for the duration of the session. The object stores information about the transactions occurring between the client and the server (Col. 4, L. 5 to 9) and is shared among servlets (Col. 4, L. 23). In terms of organizing sessions, Bayeh is explicit in stating that [w]hen session services are provided at these clustered servers, the group of sessions can logically be thought of as a *"pool, or cluster, of sessions. Since the Session Tracking facility in the Java Web Server Toolkit is only valid within the scope of a single Web server, the pool of sessions cannot be properly maintained among a group of clustered Web servers such as this."* (Col. 4, L. 59-65).

For example, if a client request is received at one server, and that server maintains information about the on-going session, there is no way for this version of the session information to be made available to a different server in the cluster if the next request from this client goes to a different server. (Col. 4, L. 59 to Col. 5, L. 3).

Thus, Bayeh teaches that sessions are identified by session IDs, session state information is maintained as session objects, the objects are shared among servlets which handle session requests, and that in a clustered server environment, one server might handle a request for a particular session and then another server might handle a subsequent request for that session. In other words, the particular session is not associated with a particular server because different servers might handle the requests for that session.

Bayeh is concerned with the discovery and retrieval of the appropriate session object in handling a request. Because Bayeh is a clustered server environment, when a request arrives which is part of a session, since the environment is not arranged to route the request to a particular server to deal with the request, the server receiving the request needs to be provided with the session object which is relevant to that session. Therefore, "[t]*he servlet engine configured as a session server further comprises a subprocess for maintaining a plurality of session objects; a subprocess for locating one of the session objects in response to a request from one of the servlets or one of the session clients; and a subprocess for returning the located session object in response to the request.*" (Col. 5, L. 65 - Col. 6, L. 4). This is not the same as in Appellant's invention, where the sessions are grouped into a plurality of groups and a "thread" is assigned to each "group" of sessions, and the **assigned thread only handles the events of the group of sessions**.

Bayeh also discloses, a load-balancing host (59) functions as a type of front-end processor to these servers, receiving client requests (100, 101, 102) and then routing those requests (shown here as 110, 111, 112) to a server selected according to policies implemented in the load-balancing host software. Note that the requests (110, 111, 112) are shown being sent to specific Web servers: this is merely an example of a possible outcome of the load balancing process. Load-balancing techniques are known in the art, and do not form part of the present invention. (Col. 8, L. 49-58.) Load balancing happens on an arbitrary basis as far as sessions are concerned which of course is not significant because the whole point of Bayeh is

that irrespective of which server receives a request, the session object can be made available.

The nature of the servlets, and the fact that they are not assigned to any particular session is underlined when one considers subsequent parts of Bayeh:

> *If a servlet is written not to use session-related information, the servlet will perform its specific processing, eventually finishing and returning its results to the server through the response object with which it was invoked. While the servlet may have been invoked with a valid session ID present in the request object, it will not have made use of that session. (Col. 10, L. 54-60.)*

and

> *If session services are required for this servlet's application, then the session services features of the present invention are used. According to the present invention, a servlet in this scenario will include code to get the session object for this session, and update that object to reflect session information related to the servlet's processing. When the servlet processing is finished, the session object is returned to the session pool, where it can be accessed for subsequent transactions with this servlet or a different servlet in the clustered environment. In this way, the state of the session can be communicated among the clustered servers and their servlets. (Col. 10, L. 64 - Col. 11, L. 8.)*

The features of Appellant's invention as recited in claim 22 are different from Bayeh, because in Appellant's invention the

sessions are between a plurality of terminals and a server. The server has a plurality of threads. The sessions are grouped into a plurality of groups and a "thread" is assigned to each "group" of sessions. Each assigned thread only handles the events of the group of sessions. This is not disclosed or suggested by Bayeh. Thus, claim 22 cannot be anticipated by Bayeh.

Further, the Examiner's argument that the "servlets" of Bayeh are the same as the "threads" claimed by Appellant in incorrect. "Servlets" and "threads" are not the same. A servlet is defined as an "applet that runs on a server." (Newton's Telecom Dictionary, 18th Edition by Harry Newton (2002), page 662; Attached hereto as Exhibit A). The term usually refers to a Java applet that runs within a Web server environment. An "applet" is a mini-program "that can be downloaded quickly and used by any computer equipped with a Java-capable browser." (Newton's Telecom Dictionary, 18th Edition by Harry Newton (2002) at page 57; Attached hereto as Exhibit B). A "thread" on the other hand is defined as "a sequence of computing instructions that makes up a process or program." (Newton's Telecom Dictionary, 18th Edition by Harry Newton (2002) at page 747, definition 1; Attached hereto as Exhibit C). This difference between a servlet and a thread is supported by Bayeh itself, for example Bayeh described the "plug-in servlet engine" as "executable code that extends the functionality of the Web server" (Col. 8, L. 63-64). Also, at column 12, lines 46-48 Bayeh describes the use of threads in a servlet. Servlets are able to handle concurrent requests, which is done by using multiple threads. A servlet is a set of machine instructions

and individual ones of the instructions can be executed by different threads.

A "thread", is essentially a placeholder information associated with a single use of a program that can handle multiple concurrent users. From the program's point of view, a thread is the resource needed to serve one individual user or a particular service request. If multiple users are using the program or concurrent requests from other programs occur, a thread is created and maintained for each of them. A program can be single-threaded or multi-threaded. A single-threaded application program insists that only a single instruction can be executed at a given time. All the instructions must be executed in an exact sequence, from the beginning to end. For example, a single-threaded Internet experience might involve your accessing a Web-based sever that would accept your request to establish a session, and would accept and serve your request for information. During this period of time, a tightly choreographed series of steps would take place in exact sequence, and no requests from other clients would be accepted until your request was satisfied. In other words, a single-threaded program must follow a single line of logic in a very rigid manner.

A multi-threaded process has multiple threads, each executing independently and each perhaps executing on separate processors within one or multiple computers. A multi-threaded program has multiple points of execution (one per thread) and, therefore can perform multiple tasks associated with multiple processes and multiple programs supporting multiple users at any given time. As each task associated with each task associated with each

process supporting each program and each user is completed, the thread for that task is resumed at the same point it had been interrupted, much as though it had been book-marked. As multiple instruction sets can be executed concurrently, the throughput and speed of running the program is much improved. In other words, a multi-threaded program, if running on a computer with multiple processors, will run much faster than a single-threaded program running on a single processor machine.

Thus, a "servlet" and "thread" are not the same and there is no teaching in Bayeh that servlets are interchangeable with threads. In fact, in column 12 lines 46-51, Bayeh discusses them as separate entities, "servlet threads" and the "servlet process". Therefore, claim 22 is patentable over Bayeh.

B. 35 U.S.C. 103(a)

1. Claim 6

Claim 6 is dependent on claim 1 and is allowable at least for the reasons noted before with respect to claim 1. Further, claim 6 recites a session is put into a first group in a first time period before suspension and put into a second group in a second time period following resumption. The Examiner notes that Bayeh "does not specifically state a session is put into a first group in a first time period before suspension and put into a second group in a second time period following resumption" but claims it would be obvious to one of ordinary skill in the art to modify Bayeh to achieve what is claimed by Applicant.

In order to establish a *prima facie* case of obviousness under 35 U.S.C. 103(a), there must be some suggestion or motivation,

whether in the references themselves or in the knowledge generally available to one of ordinary skill in the art, to modify the references or combine reference teachings. There must also be a reasonable expectation of success, and the references, when combined, must teach or suggest all of the claim limitations (See M.P.E.P. § 2142). Bayeh does not teach that the sessions are put into a first or second group or that the sessions are suspended or resumed as recited in claim 6. The Examiner suggests that Bayeh discloses that the sessions are grouped via the load balancing host software at column 8, lines 50-55. Load balancing software does not group the sessions as claimed by Appellant. Load balancing software is merely distributing processing and communications activity evenly across a computer network so that no single device is overwhelmed. If one server starts to get swamped, requests are forwarded to another server with more capacity. Thus, load balancing of Bayeh is an overflow tool that passes requests from one server to another server when the requests are to numerous for handling by a single server. This is not the same as "a session is put into a first group in a first time period before suspension and put into a second group in a second time period following resumption".

If, as the Examiner argues, one request of Bayeh is stopped and then resumed as a new session connection, the request will merely be routed to a different server if the server that initially handled the request is at maximum processing capacity. This is not taking the request from a first group and putting it into a second group but rather it is arbitrarily giving the request to a server with the required processing capacity. Nowhere is it disclosed or suggested in Bayeh that this overflow

control puts the request in a "second group". Further, there is simply no disclosure or suggestion in Bayeh of suspending or resuming the requests. Therefore, because Bayeh does not teach or suggest all of the claim limitations of claim 6, a *prima facie* case of obviousness has not been established. Thus, claim 6 is patentable over Bayeh.

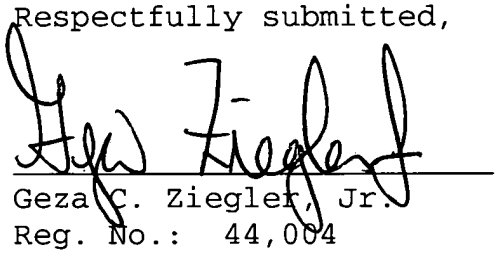
2. Claim 14

Claim 14 is ultimately dependent on claim 1 and is allowable at least for the reasons noted before with respect to claim 1. Further, claim 14 recites the terminals comprise cellular telephones. Bayeh does not disclose or suggest this feature.

In order to establish a *prima facie* case of obviousness under 35 U.S.C. 103(a), there must be some suggestion or motivation, whether in the references themselves or in the knowledge generally available to one of ordinary skill in the art, to modify the references or combine reference teachings. There must also be a reasonable expectation of success, and the references, when combined, must teach or suggest all of the claim limitations (See M.P.E.P. § 2142). Bayeh discloses a technique, system and computer program for maintaining session information among multiple clustered computers for servlets and providing those servlets with various session services (Abstract). The session services are implemented using a plug-in servlet engine (Abstract). Bayeh simply does not disclose or suggest the use of a cellular telephone. Therefore, because Bayeh does not teach or suggest all of the claim limitations of claim 14 a *prima facie* case of obviousness has not been established. Thus, claim 14 is patentable over Bayeh.

A check in the amount of $500 is enclosed herewith for the appeal brief fee. The Commissioner is hereby authorized to charge payment for any additional fees associated with this communication or credit any over payment to Deposit Account No. 16-1350.

Respectfully submitted,



8 Feb 2006

Geza C. Ziegler, Jr.
Reg. No.: 44,004
Date

Perman & Green, LLP
425 Post Road
Fairfield, CT 06824
(203) 259-1800
Customer No.: 2512

CERTIFICATE OF MAILING

I hereby certify that this correspondence is being deposited with the United States Postal Service on the date indicated below as first class mail in an envelope addressed to the Board of Patent Appeals and Interferences, United States Patent and Trademark Office, P.O. Box 1450, Alexandria, VA 22313-1450,

Date: Feb. 8, 2006

Signature: Meghan Bayne
Person Making Deposit

VIII. CLAIM APPENDIX

The texts of the claims involved in the appeal are:

1. A method of managing a plurality of sessions (66) the sessions being between a plurality of terminals (2) and a server (20) having a plurality of threads (74), the method comprising:

grouping the sessions into a plurality of groups (72); and

assigning a thread (74) to each group (72) of sessions so that the assigned thread (74) only handles the events of that group of sessions.

2. A method according to claim 1 in which grouping occurs when a session is created (70).

3. A method according to claim 1 in which grouping occurs when a session becomes active.

4. A method according to claim 1 in which one group (72) is provided for each thread (74) so that there are equal numbers of groups (72) and threads (74).

5. A method according to claim 1 in which sessions are assigned statically to particular threads (74).

6. A method according to claim 1 in which a session is put into a first group in a first time period before suspension and put into a second group in a second time period following resumption.

7. A method according to claim 6 in which the second group is chosen on the basis of the relative levels of activity of the groups.

8. A method according to claim 6 in which the second group is chosen randomly.

9. A method according to claim 1 in which each group (72) has a queue (80) and each session puts its events into that queue (80).

10. A method according to claim 1 in which the sessions are grouped by a thread referred to as an acceptor thread (76).

11. A method according to claim 10 in which the acceptor thread (76) calls a function which is answered by notification that a new session has been created and then assigns the new session to a particular session group (72).

12. A method according to claim 1 in which the sessions remain open for an undetermined period of time until closed.

13. A method according to claim 1 in which the terminals (2) comprise mobile terminals.

14. A method according to claim 13 in which the terminals (2) comprise cellular telephones.

15. A method according to claim 1 in which load balancing means is included in the assignment mechanism of the session.

16. A method according to claim 1 in which the sessions (66) involve obtaining information or conducting transactions through the Internet.

17. A method according to claim 1 in which the sessions are part of the Wireless Session Protocol (WSP).

18. A server (20) for managing a plurality of sessions with a plurality of terminal (2) the server (20) comprising a plurality of threads (74), grouping means to group the sessions into a plurality of groups and assigning means to assign a thread to each group of sessions so that the assigned thread (74) only handles the events of that group (72) of sessions.

19. A server (20) according to claim 18 comprising a gateway server serving a plurality of mobile terminals (2).

20. A server (20) according to claim 19 comprising a WAP-HTTP gateway.

21. A communications system comprising a server (20) and a plurality of terminals (2) the server (20) and the terminals (2) conducting a plurality of sessions (66) the server comprising a plurality of threads (74), grouping means to group the sessions into a plurality of groups and assigning means to assign at least one thread to each group of sessions so that the assigned thread (74) only handles the events of that group (72) of sessions.

22. A computer program product for managing a plurality of sessions (66) the sessions being between a plurality of terminals (2) and a server (20) having a plurality of threads (74), comprising:

computer readable program means for grouping the sessions (66) into a plurality of groups (72); and

computer readable program means for assigning a thread to each group (72) of sessions so that the assigned thread (74) only handles the events of that group (72) of sessions.

IX. EVIDENCE APPENDIX

1. Exhibit A - Definition of "servlet" (Newton's Telecom Dictionary, 18th Edition by Harry Newton (2002), page 662).

2. Exhibit B - Definition of "applet" (Newton's Telecom Dictionary, 18th Edition by Harry Newton (2002) at page 57).

3. Exhibit C - Definition of "thread" (Newton's Telecom Dictionary, 18th Edition by Harry Newton (2002) at page 747, definition number 1).

EXHIBIT A

NEWTON's TELECOM DICTIONARY

that enable flexible information interactions between an end user and the network.

Services On Demand An AT&T term for the immediate provision of almost any network service through universal ports, whenever required by a user; as opposed to provision via an expensive, time consuming, inflexible service order process.

Serving Area Interface A serving area interface is part of a phone company's outside plant. It is a fancy name for a box on a pole, a box attached to a wall or a box in the ground that connects the phone company's feeder or subfeeder cables (those coming from the central office) to the drop wires or buried service wires that connect to the customer's premises. It's also called a cross-wire box. See also Feeder Plant and Drop Wire.

Serving Closet The general term used to refer to either a riser or a satellite closet; Satellite Cabinet; Satellite Closet.

Serving Mobile Data Intermediate System A cellular radio term. The CDPD network entity that operates the Mobile Serving Function. The serving MD-IS communicates with and is the peer endpoint for the MDLP connection to the M-ES.

Serving Office An office of AT&T or its Connecting or Concurring Carriers, from which interstate communications services are furnished.

Serving Wire The term for the phone number that serves the location, referring to the phone number and terminating wire as one unit. Usually applies to a POTS number.

Serving Wire Center The wire center from which service is provided to the customer.

Servlet An applet that runs on a server. The term usually refers to a Java applet that runs within a Web server Web server environment. This is analogous to a Java applet that runs within a Web browser browser environment. Java servlets are becoming increasingly popular as an alternative to CGI programs. The biggest difference between the two is that a Java applet is persistent. This means that once it is started, it stays in memory and can fulfill multiple requests. In contrast, a CGI program disappears once it has fulfilled a request. The persistence of Java applets makes them faster because there's no wasted time in setting up and tearing down the process.

Servo Short for servomechanism. Devices which constantly detect a variable, and adjust a mechanism to respond to changes. A servo might monitor optical signal strength bouncing back from a disc's surface, and adjust the position of the head to compensate.

SERVORD Service Order.

SES 1. Satellite Earth Stations.

2. Severely Errored Second. A second in which a severe number of errors are detected over a digital circuit. Each error comprises a code violation (CV), such as a bipolar violation. The specific definition of SES depends on the type of circuit involved, e.g. T-1, T-3, OC-3 and OC-48. See also CV and ES.

3. Source End Station: An ATM termination point, which is the source of ATM messages of a connection, and is used as a reference point for ABR services. See DES.

Sesame Secure European System for Applications in a Multivendor Environment. Developed by the ECMA (European Computer Manufacturers Association), it is intended for very large networks of disparate origin.

Session 1. A set of transmitters and receivers, and the data streams that flow between them. In other words, an active communication, measured from beginning to end, between devices or applications over a network. Often used in reference to terminal-to-mainframe connections. Also a data conversation between two devices, say, a dumb terminal and a mainframe. It may be possible to have more than one session going between two devices simultaneously.

2. As defined under the Orange Book, a recorded segment of a compact disc which may contain one or more tracks of any type (data or audio). The session is a purely logical concept; when a multisession disc is mounted in a multisession CD-ROM player, what the user will see is one large session encompassing all the data on the disc.

Session Description Protocol See SDP.

Session Initiation Protocol See SIP.

Session key A digital key that is created by the client, encrypted, and sent to the server. This key is used to encrypt data sent by the client. See also Certificate, Digital Signature and Key Pair.

Session Layer The fifth layer — the network processing layer — in the OSI Reference Model, which sets up the conditions whereby individual nodes on the network can communicate or send data to each other. The session layer is responsible for binding and unbinding logical links between users. It manages, maintains and controls the dialogue between the users of the service. The session layer's many functions include network gateway communications.

Session Lead-In The data area at the beginning of a recordable compact disc is left blank for the disc's table of contents. The session lead-in uses up 6750 blocks space. See Track.

Session Lead-Out The data area at the end of a session which indicates that end of the data has been reached. When a session is closed, information about its con is written into the disc's Table of Contents, and the lead-out and the pre-gap are written prepare the disc for a subsequent session. The lead-out and the pre-gap together take 4650 two-kilobyte blocks (nine megabytes). See Track.

Set 1. Set is another name for a telephone.

2. SET. Secure Electronic Transaction. A developing, open specification for handling cr card transactions over any sort of network, with emphasis on Internet and the World W Web. Rather than providing the merchant with a credit card number, you send the inf mation to them in encoded form. The merchant can't see the encrypted credit card nu ber, but can forward the transaction request to the credit card company or clearinghou where the information is decrypted and verified. Only the financial institution has the k to unlock the encrypted account information. The financial institution responds to the m chant request with a digital certificate which serves to verify the authenticity of the part and the overall legitimacy of the transaction. SET ensures that no one else (e.g., hacker can gain access to your credit card information. It also ensures that an unscrupulous ecom merce merchant can't take advantage of your credit card number. The theory is that yo don't always know with whom you are dealing in Cyberspace.

Set Associative Mapping A caching technique where each block of ma computer memory is assigned to a location in each cache set where the cache is divide into multiple sets.

Set Copy Set Copy allows the duplication of programming settings from one telephone to another.

Set Top Box STB. The electronics box which sits on top of your TV, connecting it to your incoming CATV or MMDS (Microwave Multi-point Distribution System) TV signal and your TV's incoming coaxial cable. Set-tops, or converter boxes, vary greatly in their com plexity, with older models merely translating the frequency received off the cable into a fr quency suitable for the television receiver while newer models can be addressable with a unique identity much like a telephone or a node on a computer network. That identity can be addressed from the cable headend. This allows the CATV operator to turn individual channels on and off, such as pay channels.

SETA SouthEastern Telecommunications Association, a user group.

SETI Search for ExtraTerrestrial Intelligence. A federally funded project which uses arrays of radiotelescopes to search the heavens for signs of intelligent life as evidenced by radio trans missions. SETI is based on rationale laid out in a "Nature" article by physicists Philip Morrison and Guiseppe Cocconi and first implemented by Frank Drake, a Cornell astronomer. As radio waves propagate infinitely at the speed of light in the pure vacuum of space, the thinking is that at least traces of intelligent life can be identified, even though they will be of millennia long past since other stars and galaxies are thousands or millions of light years away. Promoted as a far less expensive technique than that of space travel, the project is interesting but in jeop ardy as results have been nil over the last 25 years or so. See SETI@home.

SETI@home Search for ExtraTerrestrial Intelligence at home. A project sponsored over the Internet by the Planetary Society and the University of California at Berkley. The proj ect harnesses home computers to sift through the billions of radio signals from the cosmos that pass the Earth each day in the hope of finding signals that have emanated from intel ligent life on other planets. The PCs download the program and run it against a record of signals detected by the Aricebo radiotelescope in Puerto Rico. As each PC works on the analysis, a screen saver of sorts displays a 3-D graph charting its progress. Once the anal ysis is completed, the results are uploaded to UC Berkley over the Internet, and another set of data is downloaded. http://planetary.org or http://setiathome.ssl.berkley.edu. See also SETI.

SF 1. Single Frequency. A method of inband signaling. Single frequency signaling typical ly uses the presence or absence of a single specified frequency (usually 2,600 Hz). See Signaling.

2. SuperFrame: A DS1 framing format in which 24 DS0 timeslots plus a coded framing bi are organized into a frame which is repeated 12 times to form the superframe.

SFBI Shared Frame Buffer Interconnect, a specification that makes it possible for hard ware manufactures to produce a single-board video-graphics adapter for the PC.

SFC Switch Fabric Controller.

SFD Start Frame Delimiter. A binary pattern at the end of eight octets of timing inform

EXHIBIT B

NEWTON's TELECOM DICTIONARY

and operation of all Macintosh applications are similar.

Apple Menu The Apple icon in the upper left hand corner of the Apple Macintosh screen. The Apple menu contains aliases, control panels, the chooser and other desk accessories.

Apple Pie Both an American icon, and the name chosen for Apple Computer's Personal Interactive Electronics (PIE) division, chartered with extending the company into new growth areas such as Personal Digital Assistants (PDAs), e.g. the Apple Newton. The PIE division includes Apple Online Services, Newton and Telecommunications group, publishing activities, and ScriptX-based multimedia PDA development.

Apple Remote Access ARA is Apple Computer's dial-in client software for Macintosh users allowing remote access to Apple and third party servers.

Apple URP Apple Update Routing Protocol. The network routing protocol developed by Apple for use with Appletalk.

AppleShare Apple Computer's local area network. It uses AppleTalk protocols. AppleShare is Apple system software that allows sharing of files and network services via a file server in the Apple Macintosh environment. See AppleTalk.

Applet Mini-programs that can be downloaded quickly and used by any computer equipped with a Java-capable browser. Applets carry their own software players. See Java.

AppleTalk Apple Computer's proprietary networking protocol for linking Macintosh computers and peripherals, especially printers. This protocol is independent of what network it is layered on. Current implementations exist for LocalTalk (230.4 Kbps) and EtherTalk (10 Mbps).

AppleTalk Zone and Device Filtering Provides an additional level of security for AppleTalk networks. On AppleTalk networks, network managers can selectively hide or show devices and/or zones to ARA clients. See ARA.

Application A software program that carries out some useful task. Database managers, spreadsheets, communications packages, graphics programs and word processors are all applications.

Application Based Call Routing In addition to the traditional methods of routing and tracking calls by trunk and agent group, the latest Automatic Call Distributors route and track calls by application. An application is a type of call, for example, sales or service. Tracking calls in this manner allows accurately reported calls, especially when they are overflowed to different agent groups. See ACD.

Application Binary Interface ABI. The rules by which software code is written to operate specific computer hardware. Application software, written to conform to an ABI, is able to be run on a wide variety of system platforms that use the computer hardware for which the ABI is designed.

Application Bridge Aspect Telecommunications' ACD to host computer link. Originally it ran only over R2-232 serial connections, but it now runs over Ethernet, using the TCP/IP link protocol. See also Open Application Interface.

Application Class An SCSA term. A group of client applications that perform similar services, such as voice messaging or fax-back services.

Application Entity A cellular radio term. An Application Entity provides the service desired for communication. An Application Entity may exist in an M-ES (Mobile End System) (i.e., mobile application entity) or an F-ES (Fixed End System). An Application Entity is named with an application-entity title.

Application Equipment Module AEM. A Northern Telecom term for a device within the Meridian 1 Universal Equipment Module that supports Meridian Link Modules. The Meridian Link Module (MLM) is an Application Module, specially configured to support the Meridian Link interface to host computers.

Application For Service A standard telephone company order form that includes pertinent billing, technical and other descriptive information which enables the company to provide communications network service to the customer and its authorized users.

Application Framework This usually means a class library with a fundamental base class for defining a complete program. The framework provides at least some of the facilities through which a program interfaces with the user, such as menus and windows, in a style that is internally consistent and abstracted from the specific environment for which it has been developed.

This is an explanation I received from Borland. I don't quite understand it, yet. An application framework is an object-oriented class library that integrates user-interface building blocks, fundamental data structures, and support for object-oriented input and output. It defines an application's standard user interface and behavior so that the programmer can concentrate on implementing the specifics of the application. An application framework allows developers to reuse the abstract design of an entire application by modeling each major component of an applications as an abstract class.

Application Gateway A firewall that applies security mechanisms to specific applications, such as FTP and Telnet servers. An application gateway is very effective but can impose a performance degradation.

Application Generator AG. A program to generate actual programming code. An applications generator will let you produce software quickly, but it will not allow you the flexibility had you programmed it from scratch. Voice processing "applications generators," despite the name, often do not generate programming code. Instead they are self-contained environments which allow a user to define and execute applications. They are more commonly called applications generator, since one generator can define and execute many applications. See Applications Generator for a longer explanation.

Application Layer The topmost, visible to the user, presentation of a communications network; the user interface point in network architectures. See Open Systems Interconnection — Reference Model.

Application Level Firewall A firewall system in which service is provided by processes that maintain complete TCP connection state and sequencing. Application level firewalls often re-address traffic so that outgoing traffic appears to have originated from the firewall, rather than the internal host.

Application Level proxy A firewall technology that involves examining application specific data in order to guard against certain types of improper or threatening behaviors.

Application Metering The process of counting the number of executions of the copies of an application in use on the network at any given time and ensuring that the number does not exceed preset limits. Application metering is usually performed by a network management application running on the file server. Most application metering software will allow only a certain number of copies (usually that number specified in the application software license) of an application to run at any one time and will send a message to any users who try to exceed this limit.

Application Module A Northern Telecom term for a computer that can be attached to a Northern Telecom phone system and add intelligence and programmability to the phone system. Often, the AM will be a computer conforming to open standards, such as DOS or Windows, or it may be VME-based.

Application Module Link AML. A Northern Telecom internal and proprietary link that connects the Meridian 1 (via EDSI or MSDL port) to the Meridian Link Module.

Application Program A computer software program designed for a specific job, such as word processing, accounting, spreadsheet, etc.

Application Program Interface API. A set of formalized software calls and routines that can be referenced by an application program to access underlying network services.

Application Programming Interface API. A set of functions and values used by one program to communicate with another program or with an operating system. See API for a better explanation.

Application Profile As SCSA term. A description of the kinds of resources and services required by a client application (or an application class). An application profile is defined once for an instance of an application; then system services such as the SCR will be able to fulfill the needs of the application without the application having to state its needs explicitly.

Application Server A dedicated, heavy duty PC which sits on a corporate network and contains a program which people on the network share. Such program would typically be a database — perhaps a sales force automation program, such as Goldmine, Act or Maximizer. See also Database Server.

Application Service Element ASE. A messaging term. A module or portion of a protocol in the application layer 7 of the OSI (Open Systems Interconnection) protocol stack. Several ASEs are usually combined to form a complete protocol, e.g., the X.400 P1 protocol which consists of the MTSE (Message Transfer Service Element), and the RTSE (Reliable Transfer Service Element).

Application Service Provider ASP. The definition of an ASP is evolving. Today it's a company which offers software to business users over the Internet on some sort of per-use charge. For business users, an ASP is a kind of outsourcer; users are not required to buy, own or take care of their own software. Instead of buying software, buying the heavy duty computers to run it on and the heavy duty broadband telecommunications network to get it to all their distant corporate users, the user companies (i.e. the

EXHIBIT C

NEWTON's TELECOM DICTIONARY

Thin Computing See Thin Client.

Thin Ethernet A coaxial (0.2-inch, RG58A/U 50-ohm) that uses a smaller diameter coaxial cable than standard thick Ethernet. Thin Ethernet is also called "Cheapernet" due to the lower cabling cost. Thin Ethernet systems tend to have transceivers on the network interface card, rather than in external boxes. PCs connect to the Thin Ethernet bus via a coaxial "T" connector. Thin Ethernet is now the most common Ethernet coaxial cable, though twisted pair is gaining. Thin Ethernet is also referred to as ThinNet, ThinWire or Cheapernet. See also 10BASE-T.

Thin-Film Interference Filter Thin-film filters control the reflection, refraction, transmission, and absorption of light waves. Filters are used in a wide variety of optical components, and the majority of today's WDM systems incorporate thin-film filter technology for multiplexing and demultiplexing. Interference filters are constructed by depositing a series of coatings with different refractive indexes on a glass substrate. This construction generates interference patterns as lightwaves pass through such that certain wavelengths are reflected while others pass through undisturbed. In DWDM applications, at lower channel counts and larger spacing between wavelengths, thin-film filters are produced in volume quantities that can process channel spacings of 200 Ghz. However, at higher channel counts and smaller channel spacings (below 100 Ghz and 50 Ghz), manufacture becomes increasingly difficult. The value proposition for competing multiplexing/demultiplexing technologies such as arrayed waveguides and fiber bragg gratings becomes increasingly compelling as channel count increases.

Thingy See Dingy.

Thinnet Jargon used to describe thin Ethernet coaxial cable. Referred to ThinNet, ThinWire or Cheapernet.

Thinwire The 50-ohm coaxial cable listed in IEEE 802.3 specifications and used in some Ethernet local area network installations.

Third Generation Wireless See 3G.

Third Order Harmonics The third multiple of a specific frequency or a specific frequency multiplied by three.

Third Party Call Any call charged to a number other than that of the origination or destination party. It's not a good idea to let your employees make third party calls to one or more of your phone numbers. Best to ask them to place the calls on their personal phone credit cards. This way, they will spend a modicum of time justifying their exorbitant phone calls.

Third Party Call Control A call comes into your desktop phone. You can transfer that call. When the phone call has left your desk, you can no longer control it. That is called First Party Call Control. If you were still able to control the call (and let's say, switch it elsewhere) that would be called Third Party Call Control. Some Computer Telephony links allow only first party call control. Some allow third party as well. If you control the switch — the PBX or the ACD — you will typically have Third Party Call Control. If you just control the desktop, you'll typically have only First Party Call Control. There is no such animal as Second Party Call Control.

Third Party Cookie See Cookie.

Third Party Verification TPV. Before your Primary Interexchange Carrier (PIC), or long distance carrier, can be changed, the FCC now (1988) requires that the new carrier have a in place a means of verifying that you have authorized such a change. Business customers must execute a written Letter of Agency (LOA), which the new carrier can present to the old carrier. The veracity of a change for residential customers is ensured through Third-Party Verification (TPV), which takes the following form. Once you have concluded your conversation with the sales representative of the new carrier, the sales rep will initiate a conference call to add to the call a representative of an independent third party. The third party will verify the change, including all relevant information recorded by the sales representative. This step protects you from "slamming," which is the practice of changing your PIC without your authorization. See also LOA, PIC, and Slamming.

Third Wire Tap The activating of a telephone handset microphone by using a third wire, thus bypassing the hook switch.

THL Trans Hybrid Loss.

Thought Police In Imperial Japan before World War II, members of the "thought police" — Shiso Keisatu — spread out to suppress dangerous thinking in the populace. Such dangerous thinking was obviously different to what the imperial powers wanted. The Thought Police were disbanded by General McArthur when he imposed freedom of speech after the War. The thought police was later chillingly immortalized by George Orwell in his 1949 "1984."

Thousand Block Number Pooling See Number Pooling.

Thread 1. A thread is a sequence of computing instructions that makes up a process or program. A program can be single-threaded or multi-threaded. A single-threaded application program insists that only a single instruction can be executed at a given time. All the instructions must be executed in an exact sequence, from beginning to end. For example, a single-threaded Internet experience might involve your accessing a Web-based server that would accept your request to establish a session, and would accept and serve your request for information. During this period of time, a tightly choreographed series of steps would take place in exact sequence, and no requests from other clients would be accepted until your request was satisfied. In other words, a single-threaded program must follow a single line of logic in a very rigid manner.

A multi-threaded process has multiple threads, each executing independently and each perhaps executing on separate processors within one or multiple computers. A multi-threaded program has multiple points of execution (one per thread) and, therefore can perform multiple tasks associated with multiple processes and multiple programs supporting multiple users at any given time. As each task associated with each task associated with each process supporting each program and each user is completed, the thread for that task is resumed at the same point it had been interrupted, much as though it had been bookmarked. As multiple instruction sets can be executed concurrently, the throughput and speed of running the program is much improved. In other words, a multi-threaded program, if running on a computer with multiple processors, will run much faster than a single-threaded program running on a single processor machine.

2. In the context of an Internet Usenet newsgroup or other interactive discussion forum, a thread essentially is a train of thought or line of logic that can be followed through the fabric of a larger subject. A thread begins with a message posting. Responses are "hung off" of that initial posting in a chronological and hierarchical fashion as comments are offered and elicit additional comments, and as questions are posed and answered and the answers elicit yet other sets of questions and answers. This hierarchical threading is very easy to follow in graphical format, such as that used in the World Wide Web.

Threaded A method of presenting articles within a newsgroup in a way that shows which articles refer to which other ones. See Thread.

Threaded Code Threaded code is also known as Threaded Pseudo Code, or threaded p-code. It was first popularized in a software language called Forth. Eventually Microsoft fixed Forth and changed it into Basic. See Thread.

Threat Analysis Examination of all actions and events that might adversely affect a system, a network or an operation.

Three Finger Salute Ctrl Alt Delete.

Three Dog Night Three Dog night, attributed to Australian Aborigines) came about because on especially cold nights these nomadic people needed three dogs (dingos, actually) to keep from freezing.

Three Nines 99.9%. Three nines typically refers to the reliability of a system (computer, telephone system, etc.) that works 99.9% of the time. These days the industry talks increasingly of five nines reliability, i.e. 99.999% and fo six times reliability, e.g. 99.9999%.

Three Slot An obsolete pay phone that is identified by three separate coin slots.

Three-Tier A type of client/server architecture consisting of three well-defined and separate processes, each running on a different platform:

1. The user interface, which runs on the user's computer, also called the client.
2. The functional modules that process the data. This middle tier runs on a server. It is often called the application server.
3. A database management system (DBMS) that stores the data required by the middle tier. This tier runs on a second server called the database server.

The three-tier design has some advantages over traditional two-tier or single-tier designs. Its modularity makes it easier to change or replace one tier without affecting the other tiers. It's also better for load balancing.

Three-Way Calling A local phone company feature that allows a phone user to add another user to an existing conversation and have a three party conference call.

Three-Way handshake The process whereby two protocol entities synchronize during connection establishment.

Three-Watt Booster Optional equipment for use with a cellular phone car-mounting kit that raises a portable phone's maximum transmission power from 0.6 watts to 3.0 watts.

Three-Way Conference Transfer A PBX feature. By depressing the switch

X. RELATED PROCEEDINGS APPENDIX

Not applicable.

This Page is Inserted by IFW Indexing and Scanning Operations and is not part of the Official Record

BEST AVAILABLE IMAGES

Defective images within this document are accurate representations of the original documents submitted by the applicant.

Defects in the images include but are not limited to the items checked:

- ☐ BLACK BORDERS
- ☐ IMAGE CUT OFF AT TOP, BOTTOM OR SIDES
- ☐ FADED TEXT OR DRAWING
- ☐ BLURRED OR ILLEGIBLE TEXT OR DRAWING
- ☐ SKEWED/SLANTED IMAGES
- ☐ COLOR OR BLACK AND WHITE PHOTOGRAPHS
- ☐ GRAY SCALE DOCUMENTS
- ☐ LINES OR MARKS ON ORIGINAL DOCUMENT
- ☐ REFERENCE(S) OR EXHIBIT(S) SUBMITTED ARE POOR QUALITY
- ☐ OTHER: _____

IMAGES ARE BEST AVAILABLE COPY.

As rescanning these documents will not correct the image problems checked, please do not report these problems to the IFW Image Problem Mailbox.